Named Entity Recognition for Entity Linking: What Works and What's Next

Simone Tedeschi[1], **Simone Conia**[2], **Francesco Cecconi**[1] and **Roberto Navigli**[2]
[1]Babelscape, Italy
[2]Sapienza NLP Group, Sapienza University of Rome
{tedeschi, cecconi}@babelscape.com
conia@di.uniroma1.it navigli@diag.uniroma1.it

Abstract

Entity Linking (EL) systems have achieved impressive results on standard benchmarks, mainly thanks to the contextualized representations provided by recent pretrained language models. However, such systems still require massive amounts of data – millions of labeled examples – to perform at their best, with training times that often exceed several days, especially when limited computational resources are available. In this paper, we look at how Named Entity Recognition (NER) can be exploited to narrow the gap between EL systems trained on high and low amounts of labeled data. More specifically, we show how and to what extent an EL system can benefit from NER to enhance its entity representations, improve candidate selection, select more effective negative samples and enforce hard and soft constraints on its output entities. We release our software – code and model checkpoints – at https://github.com/Babelscape/ner4el.

1 Introduction

Entity Linking (EL), also known as Named Entity Disambiguation (NED), is the task of associating an ambiguous textual mention with a named entity in a knowledge base. Indeed, named entities may have several surface forms – their full names, partial names, aliases and abbreviations – making EL a very challenging task in Natural Language Processing (NLP). Over the years, EL systems have achieved impressive results in standard benchmarks, especially thanks to the advent of modern language models (Devlin et al., 2019), and have found innumerable applications in a wide range of downstream tasks, including Information Extraction (Lin et al., 2012; Guo et al., 2013; Rao et al., 2013), Question Answering (Yin et al., 2016; Dubey et al., 2018), knowledge base population (Ji and Grishman, 2011) and recommender systems (Musto et al., 2014; Di Noia and Ostuni, 2015; De Gemmis et al., 2015), *inter alia*.

In general, EL systems are composed of two main components: a candidate generation module and a mention disambiguation module. The aim of the former is to select from a knowledge base (e.g. Wikipedia) a suitable subset of named entities that can be associated with a given textual mention in an input text. This set of candidates is then given to the latter module whose objective is to choose and assign the most appropriate entity to the mention. Recent studies (Shahbazi et al., 2019; Broscheit, 2019; Botha et al., 2020; Cao et al., 2021) have shown that learning better representations of mentions and entities is key to improving the two aforementioned components and enabling state-of-the-art results. However, one common issue with current EL approaches is that they require massive amounts of training data – often millions of labeled items – in order to perform at their best, making the development of a high-performance EL system viable only to a limited audience.

In this paper, we study whether it is possible to narrow the performance gap between systems trained on limited and large amounts of data. In particular, we take a look at Named Entity Recognition (NER) – the task of identifying specific words as belonging to predefined semantic types such as Person, Location, Organization – and how this task can be exploited to improve a strong Entity Linking baseline in low-resource settings without requiring any additional data. With this as our aim, we introduce a fine-grained set of NER classes and propose multiple approaches to the exploitation of NER for EL, showing how a state-of-the-art model can benefit from them. Our main contributions can be summarized as follows:

- We introduce new fine-grained classes for NER and use them to automatically label each entity in Wikipedia;

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 2584–2596
November 7–11, 2021.

- We show how such classes can easily be used to integrate type information into the entity representations of EL systems;
- We propose a NER-enhanced candidate generation module which decreases the size of the candidate set while increasing recall;
- We present a NER-constrained decoding module to discard unlikely outputs during mention disambiguation;
- We demonstrate how NER-based negative sampling helps a model produce more accurate entity representations at training time;
- We assess the effectiveness of our contributions on multiple standard benchmarks for EL, showing consistent improvements over strong baseline systems.

We hope that our work will provide a stepping stone for further studies on the interplay between Entity Linking and Named Entity Recognition, and encourage further studies on high-performance EL systems for scenarios in which only a small amount of labeled data is available. We release our software – code and model checkpoints – at https://github.com/Babelscape/ner4el.

2 Related Work

Entity Linking. Over the past few years, neural approaches have attained strong results in EL, especially thanks to the advances in contextualized word embedding and entity representation techniques (Ganea and Hofmann, 2017; Le and Titov, 2018, 2019; Yang et al., 2019). While initial work relied on static word embeddings such as word2vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014) to represent mentions and entities, recent studies (Shahbazi et al., 2019; Broscheit, 2019; Botha et al., 2020; Cao et al., 2021) have shown the benefit of employing contextualized embeddings from pretrained language models, such as ELMo (Peters et al., 2018), BERT (Devlin et al., 2019) and BART (Lewis et al., 2020). Notably, researchers are tackling EL with a variety of very different approaches. For example, Botha et al. (2020) put forward a dual-encoder architecture, composed of two separate encoders for mentions and entities, that maximizes the similarity between a mention embedding and its corresponding entity embedding, whereas Cao et al. (2021) proposed GENRE which, given a mention in context, generates its unique name (e.g. the title of its corresponding Wikipedia page) in an autoregressive fashion. Nevertheless, recently-proposed systems, in order to achieve high performance, require training on millions of samples – GENRE is trained on KILT (Petroni et al., 2021) which is made up of 9M training instances – and this often means days-long training times. Currently, in EL a researcher with a limited hardware budget must therefore decide between training on lower amounts of data at the cost of drastic drops in performance – the performance of GENRE drops by 8.8 points in F_1 when trained only on the AIDA-YAGO-CoNLL training set – and long training times. In this paper, instead, we show that the clever use of NER for EL can significantly narrow the gap between systems trained on thousands as opposed to millions of instances, while retaining the benefits of shorter training times.

Enriching Entity Linking. While the first successful approaches to EL often relied on non-neural graph-based techniques (Hoffart et al., 2011; Rao et al., 2013; Moro et al., 2014), there is a growing body of work that studies how to enrich neural models by taking advantage of relational knowledge from semantic networks such as Wikidata, YAGO (Suchanek et al., 2007), WordNet (Miller, 1995) and BabelNet (Navigli and Ponzetto, 2012; Navigli et al., 2021), *inter alia*. For example, Raiman and Raiman (2018) proposed DeepType which relies on Wikidata to integrate symbolic knowledge into the reasoning process of a neural network. In particular, they make use of a type system to constrain the behavior of an entity prediction model with respect to the symbolic structure defined by types. Another notable work in this direction is Bootleg (Orr et al., 2020), a system which uses the edges defined in Wikidata and YAGO to encode entity relations and entity types as input embeddings to a Transformer-based architecture. However, while there is clear evidence that integrating relational knowledge into EL approaches is beneficial, the sparsity of such relations may make them an unappealing option for low-data scenarios.

Named Entity Recognition. Similarly to almost any other area in NLP, Named Entity Recognition systems have benefited greatly from the advent of pretrained language models (Virtanen et al., 2019; Mueller et al., 2020; Liang et al., 2020; Souza et al., 2020). Nowadays, their performance makes such

systems extremely compelling options in downstream tasks such as EL. Indeed, thanks to its coarse-grained classes, NER is an obvious way to cluster entities and, therefore, to reduce the intrinsic sparsity of the Entity Linking task. However, there is a surprisingly low number of studies on the effectiveness of enriching EL models with NER information. Most of the contributions in this direction use NER as a preprocessing step before EL, or learn directly to perform the tasks jointly (Luo et al., 2015; Nguyen et al., 2016; Kolitsas et al., 2018; Martins et al., 2019; Broscheit, 2019).

In our work, we take the best of both worlds, and not only do we propose other ways to exploit NER for EL, but we also show that individual NER approaches can be combined to further improve a strong EL model.

3 NER for EL

In this Section, we take inspiration from what has already been shown to work and propose several new methods for exploiting NER for EL. To this end, we first describe a simple yet strong baseline into which we will plug our NER-focused contributions (Section 3.1). In particular, we introduce a set of finer-grained NER classes (Section 3.2) and use them to inject NER information into entity representations (Section 3.3), devise a NER-enhanced candidate generation module (Section 3.4), better select negative samples during training (Section 3.5), and introduce a NER-constrained decoding technique (Section 3.6).

3.1 Baseline System

Our baseline system for EL is composed of two main modules: a candidate generation module and a mention disambiguation module. Given an input sentence with pre-identified mentions, the former of the two modules is responsible for i) retrieving a set of candidate entities of any given mention from an alias table, and ii) reducing the size of this set by taking the top-k candidates according to their frequency in Wikipedia. The latter module is, instead, a neural architecture which features two Transformer-based encoders – one to represent a mention in context, the other to represent candidate entities – whose output states are used to assign the most appropriate entity to the considered mention.

More formally, let ϕ and ψ be the mention and entity encoders of the disambiguation module. The disambiguation module uses ϕ and ψ to compute the cosine similarity score of each mention-candidate pair (m, c_i) for each $i \in \{1, \ldots, k\}$ and selects the highest-scoring entity ε as follows:

$$\varepsilon = \underset{i \in \{1,\ldots,k\}}{\operatorname{argmax}} \frac{\phi(m)^T \psi(c_i)}{\|\phi(m)\| \|\psi(c_i)\|} \quad (1)$$

Following Botha et al. (2020), the mention encoder ϕ takes as input a sequence of tokens in which the start and the end of the mention m is identified by special tokens ([E] and [/E]) and surrounded by left and right contexts of at most 64 tokens, whereas the entity encoder ψ models each entity by taking as input the first 128 tokens of the corresponding Wikipedia article.

3.2 Fine-Grained Classes for NER

In its standard formulation, NER distinguishes between four classes of entities: Person (PER), Location (LOC), Organization (ORG), and Miscellaneous (MISC). Although NER systems that use these four classes have been found to be beneficial in downstream tasks, we argue that they might be too coarse-grained and, at the same time, not provide a sufficiently exhaustive coverage to also benefit EL, as many different entities would fall within the same MISC class.

For these reasons, we introduce a new set of finer-grained NER classes, namely, Person (PER), Location (LOC), Organization (ORG), Animal (ANIM), Biology (BIO), Celestial Body (CEL), Disease (DIS), Event (EVE), Food (FOOD), Instrument (INST), Media (MEDIA), Monetary (MON), Number (NUM), Physical Phenomenon (PHYS), Plant (PLANT), Supernatural (SUPER), Time (TIME) and Vehicle (VEHI).

We design our set of classes starting from the 18 fine-grained classes used for OntoNotes 5.0 (Pradhan et al., 2012), splitting and merging them to better fit the EL task. For example, we split the PRODUCT class of OntoNotes into three separate classes, namely FOOD, INST and VEHI, and merge the QUANTITY, ORDINAL, CARDINAL and PERCENT classes of OntoNotes into a single NUM class. We provide more details about how fine-grained NER classes are compared with the ones in OntoNotes in Appendix B.

At this stage, in order to use the newly introduced NER classes, we label each Wikipedia entity with one of them by taking advantage of WordNet, a manually-created network of synsets, and

NER Tag	NER Class	# Wikipedia articles
PER	Person	1,886K
ORG	Organization	439K
LOC	Location	1,228K
ANIM	Animal	330K
BIO	Biology	16K
CEL	Celestial Body	13K
DIS	Disease	9K
EVE	Event	249K
FOOD	Food	15K
INST	Instrument	52K
MEDIA	Media	703K
MON	Monetary	2K
NUM	Number	1K
PHY	Physical Phen.	2K
PLANT	Plant	51K
SUPER	Supernatural	6K
TIME	Time	9K
VEHI	Vehicle	78K
Total	—	5,089K

Table 1: Our new set of finer-grained classes for NER and the number of Wikipedia articles, and therefore entities, that we map to each class.

BabelNet[1], which provides a high-quality mapping between WordNet concepts and Wikipedia pages. In particular, we start by selecting and manually annotating a seed set of the 200 highest-level nominal concepts from the WordNet hypernymy taxonomy. Then, we expand this gold seed set using a breadth-first search algorithm to gradually include concepts that are linked to the seed set through hyponymy edges, thus creating a silver seed set of around 40K concepts. We repeat this process, starting from the newly created silver seed set to assign a NER class to each concept in the BabelNet graph, which also includes the concepts of WordNet. Finally, since most concepts in BabelNet are linked to Wikipedia pages, we now have a situation where each entity in Wikipedia is labeled with one of our NER classes.

Table 1 provides an overview of the number of Wikipedia articles for each NER class; we release this mapping together with our software to encourage the use of these classes not only in EL, but also in other tasks. We used BabelNet 5.0, which includes the November 2020 dump of the English Wikipedia.

[1] https://babelnet.org

3.3 NER-Enhanced Entity Representation

In the baseline system we presented in Section 3.1 for EL, the aim of the mention encoder ϕ is to produce a dense mention representation that is as similar as possible to the representation produced by the entity encoder ψ for its most appropriate entity. The better and richer the representations for the candidate entities are, the easier it will be for the system to disambiguate the corresponding mentions. One way to enrich the representation of each candidate entity is to make the entity encoder aware of class information. In particular, together with the textual description of an entity, we propose also providing the NER class as an additional input to the entity encoder. More specifically, we prepend the NER tag of a Wikipedia entity to its textual description and feed this enhanced string to the entity encoder. Not only does this feature help the entity encoder to better distinguish between entities that belong to different NER classes, but it also leads the mention encoder to consider such classes indirectly when producing the dense representation of a mention.

3.4 NER-Enhanced Candidate Generation

In the candidate generation step, the aim is to select a suitable set of candidates for each mention in context. The desired properties for such a set of candidates are high recall – target entities should as frequently as possible be within the corresponding candidate sets – but also a small number of candidates to choose from, so as to make the disambiguation step as easy as possible. However, the majority of mentions tends to have dozens of candidates – the most common mentions also being the most ambiguous, following the Zipf's law – and, therefore, in order to satisfy the second desired property, several EL systems set an upper bound to the size of the candidate set, in this way hampering candidate recall. Moreover, selecting this upper bound adds another layer of complexity to finding the best trade-off between recall and size.

In this Section, instead, we propose a strategy for considerably decreasing the size of the candidate set while also increasing its recall. Specifically, we train and employ a NER classifier to predict the NER class of an input mention in context, and then discard all the candidates whose class is different from the predicted one. For example, consider the sentence in Figure 1, where the mention *Tesla* would normally have a total of 18 candidate enti-

ties to choose from. If we limited the candidates to the 10 most popular entities, then the correct entity *Tesla (band)* would be left out, making it impossible for the system to disambiguate the input correctly. Instead, thanks to our proposed strategy, if the NER classifier predicts the correct NER class, namely ORG, all the candidates that do not correspond to organizations will be discarded from the candidate set. In our example, as we can see in Figure 1, not only does our NER-filtered candidate set include the target entity, but it is also considerably smaller (4 candidates instead of 18). Our NER classifier is a BERT-based model, which takes as input a mention in context and outputs one of the 18 classes introduced in Section 3.2.

3.5 NER-based Negative Sampling

Our baseline system learns to model the representation of a mention by comparing it with the representations of the corresponding candidate entities, both correct and wrong ones. However, some mentions are unambiguous (i.e., they have only one possible candidate entity), leading to sub-optimal learning. One common way to overcome this problem is to add negative samples. In EL, negative samples are simply entities added to the candidate set of a mention with the aim of letting the model learn more accurate mention representations which are "semantically" near to the representations of the target entities and far from the ones of the negative samples (our baseline system already makes use of them).

Although adding negative samples indiscriminately has already been proven to be beneficial for EL, we propose a more refined approach in which we select specific negative samples according to their NER class. In particular, given a mention, its target entity and its NER class c, we enlarge the candidate set at training time by adding a number of negative samples belonging to the same class c. The main motivation for using this NER-based negative sampling strategy is to make the training process more challenging and further stress the system to produce better representations. Indeed, the textual descriptions of entities belonging to the same NER class are often similar and follow recurring patterns – e.g., in Wikipedia a person is usually described by their date, place of birth and occupation – and therefore adding NER-based negative sample encourages the underlying neural network to rely on entity-specific features.

Figure 1: Example of the NER-enhanced Candidate Generation module. The *Tesla* mention has 18 candidates, and including only the 10 most popular entities in the candidate set, the target entity *Tesla (band)* would be not included. Applying our strategy, instead: i) the correct entity is included and, ii) the dimension of the resulting set is significantly smaller.

3.6 NER-Constrained Decoding

So far, we have introduced a few strategies to enhance an EL baseline by exploiting NER at the input level or during the training process. In this Section, instead, we propose a strategy that uses NER to improve our EL baseline at the output level by enforcing "soft" and "hard" constraints at infer-

ence time. Our intuition is that, for very ambiguous mentions, an EL system may be biased towards very frequent entities, independently of the context such mentions appear in. In order to mitigate this issue, we propose constraining our EL system to output an entity whose NER class is consistent with the prediction(s) of a NER classifier for the same input mention. In our experiments, we distinguish between "hard" and "soft" constraints, with the difference being that in the former we force the entity predicted by the EL system to be exactly the same as that predicted by the NER classifier, whereas in the latter we force the entity to belong to one of the top-k predictions of the NER classifier. More details are provided in Appendix C.

We also analyze two alternatives for building the NER classifier: i) training a separate model as we do in Section 3.4, or ii) training the EL system not only to assign the most appropriate entity to a mention in context but also to provide its NER class. The advantage of the second approach is that it requires just a single model and, therefore, fewer computational resources. However, performing both tasks jointly results in worse scores in NER labeling, which, in turn, decreases the benefits of our NER-constrained decoding strategy for the overall EL system (see Appendix C).

3.7 Combinations of NER Contributions

Some of our contributions can be combined to bring further improvements. For example, it is possible both to enhance entity representations (Section 3.3) and also to apply our NER-constrained decoding strategy (Section 3.6). Similarly, it is possible to add NER-based negative samples during training to let the model produce more accurate representations (Section 3.5) and also apply our decoding strategy; or even to combine all three of the above-mentioned contributions. One interesting combination consists in first removing all the candidates whose NER class is different from the one predicted for the input mention (Section 3.4), and then increasing the size of the candidate set by adding negative samples of the same class (Section 3.5), making the training process more challenging.

4 Experiments

In this Section, we describe our experimental setup (Section 4.1), the datasets we use to train and evaluate our NER-based approaches (Section 4.2), the results of each contribution (Section 4.3), followed by an analysis of the benefits of NER for EL (Section 4.4).

4.1 Experimental Setup

We implemented our NER classifier, our baseline EL model, and our NER-based enhancements for EL with PyTorch, using the Transformers library (Wolf et al., 2020) to load and fine-tune the weights of `BERT-large-uncased`. We trained each model configuration for 30 epochs, adopting an early stopping strategy with a patience value of 5, with Adam (Kingma and Ba, 2015) and a learning rate of 10^{-5}, as standard when fine-tuning the weights of a pre-trained language model. We use the same NER classifier for all experiments except for those that involve jointly learning NER and EL. Our NER classifier achieves 97.1% in terms of accuracy on the AIDA-YAGO-CoNLL test set. In the remainder of this Section, we report the results of the best model checkpoints according to their F_1 score on the validation split of the AIDA-YAGO-CoNLL dataset computed at the end of each training epoch. We provide further details about the hyperparameter values, training times and hardware infrastructure in Appendix A.

4.2 Datasets

In the following, we describe the datasets we use to train, validate and test our contributions. We stress that we train each of our model configurations on only the AIDA-YAGO-CoNLL training split, i.e., on only 18K labeled instances as opposed to the millions on which current state-of-the-art systems are trained, showing the benefits of NER when a scarce amount of labeled instances are available. While there is a growing interest in multilingual datasets for both NER (Tedeschi et al., 2021) and EL (Botha et al., 2020), in this work we focus only on the English language.

AIDA-YAGO-CoNLL (Hoffart et al., 2011) is one of the largest manually annotated EL datasets for English as it contains 388 articles with 27,817 linkable mentions corresponding to the named entities annotated for the original CoNLL-2003 entity recognition task (Tjong Kim Sang and De Meulder, 2003) This dataset comprises a number of newswire articles taken from the Reuters Corpus.

MSNBC, AQUAINT and ACE2004 are smaller evaluation sets, cleaned and updated by Guo and Barbosa (2017). MSNBC consists of 20 news articles from 10 different topics (two articles per topic)

System	training instances	AIDA accuracy
GENRE	18K	88.6
Our EL baseline	18K	88.8
Our EL baseline **+ NER**	18K	**92.5**
GENRE	9000K	93.3

Table 2: The first two rows show the InKB accuracy of our baseline system and GENRE trained, validated and tested on the AIDA-YAGO-CoNLL (in-domain setting). The last row shows the in-domain accuracy of GENRE when pretrained on KILT which is made up of 9 million training instances. Our NER-enhanced EL system is almost able to bridge the gap in performance when trained on only 18 thousand instances.

Model	AIDA
Our EL baseline	88.8
w/ NER-enhanced Representations (NER-R)	89.3
w/ NER-based Negative Sampling (NER-NS)	89.6
w/ NER-enhanced Candidate Generation (NER-CG)	89.4
w/ NER-constrained Decoding (NER-CD)	92.2
w/ NER-R + NER-NS	89.7
w/ NER-R + NER-CD	92.3
w/ NER-NS + NER-CG	90.0
w/ NER-NS + NER-CD	92.4
w/ NER-R + NER-NS + NER-CD	**92.5**

Table 3: Accuracy of our proposed NER-based contributions and their combinations on the AIDA-YAGO-CoNLL test set. Each contribution improves the performance of the baseline, and two or more NER-based approaches can be combined to further improve the results.

and 656 linkable mentions. AQUAINT is made up of 50 documents and 727 linkable mentions from the Xinhua News Service, the New York Times and the Associated Press. Finally, ACE2004 features a set of 35 news articles and 257 linkable mentions.

WNED-WIKI and WNED-CWEB are larger, but automatically extracted, evaluation sets for EL. They were built from the ClueWeb and Wikipedia corpora by Guo and Barbosa (2017) and Gabrilovich et al. (2013). WNED-WIKI, or simply WIKI, consists of 320 documents and 11,154 mentions, while WNED-CWEB, or simply CWEB, consists of 320 documents and 6,821 mentions.

4.3 Results

In what follows, we first show the overall results of our NER-enriched approaches for EL on an in-domain evaluation, and then we focus on the individual benefits of each contribution. Finally, we show that such contributions are robust and beneficial in out-of-domain evaluations.

NER for EL. As can be seen in Table 2, when trained on only the 18K instances of the AIDA-YAGO-CoNLL training split, our baseline EL system obtains results that are on par – 88.8% against 88.6% in accuracy on the test split of AIDA-YAGO-CoNLL – with those of GENRE (Cao et al., 2021) which is, on average, the current best-performing system across the datasets described in Section 4.2. Table 2 also shows that GENRE benefits greatly from drastically increasing the size of the training set from 18K to 9000K labeled instances (Petroni et al., 2021), gaining almost 5 points in accuracy. As we argued in Section 2, this improvement comes at the cost of much longer training times and/or more expensive hardware. However, if we put together our NER-focused contributions, they allow our baseline EL model to significantly narrow this gap, improving accuracy on the AIDA-YAGO-CoNLL test set by almost 4 points, while still using the original small training set with only 18K labeled instances.

What contributes to these results? One may wonder what the most important contributions are among the NER-focused approaches we propose. Table 3 reports the results of each of the contributions we described in Sections 3.3-3.7. As one can see, even the smaller contribution, i.e., enriching the representations of an entity by including its NER class (see Section 3.3), provides an improvement of 0.5 points in accuracy, while the most beneficial individual contribution is our NER-based constrained decoding strategy (Section 3.6), which provides an improvement of 3.6 points in accuracy. Moreover, we also observe that several of our contributions are complementary, in that their combinations bring further improvements, with the best combination attaining an accuracy of 92.5% on the test set of AIDA-YAGO-CoNLL.

Out-of-domain results. Finally, Table 4 shows that our NER-focused contributions bring benefits on out-of-domain evaluations too. Similarly to what we observed in the in-domain setting, our individual contributions consistently improve the results across popular out-of-domain test sets, namely, MSNBC, AQUAINT, ACE2004, CWEB and WIKI. Moreover, the combination of two or

Model	MSNBC	AQUAINT	ACE2004	CWEB	WIKI	Avg.
Our EL baseline	86.9	61.8	89.9	65.7	58.5	72.6
w/ NER-enhanced Representation (NER-R)	86.9	62.2	89.9	66.1	59.0	72.8
w/ NER-based Negative Sampling (NER-NS)	87.0	62.4	90.0	66.5	59.2	73.0
w/ NER-enhanced Candidate Gen. (NER-CG)	87.0	62.3	89.9	66.4	59.2	73.0
w/ NER-Constrained Decoding (NER-CD)	89.2	68.2	91.2	68.1	63.8	76.1
w/ NER-R + NER-NS	87.0	62.5	90.0	66.3	59.5	73.1
w/ NER-R + NER-CD	88.7	69.2	91.2	68.3	63.5	76.2
w/ NER-NS + NER-CG	87.8	65.4	90.2	66.8	60.5	74.1
w/ NER-NS + NER-CD	89.1	69.2	91.2	68.4	63.8	76.3
w/ NER-R + NER-NS + NER-CD	**89.2**	**69.5**	**91.3**	**68.5**	**64.0**	**76.5**

Table 4: InKB accuracy of our EL baseline system and of the NER-based contributions on the out-of-domain test sets of MSNBC, AQUAINT, ACE2004, WNED-CWEB and WNED-WIKI.

Ablation – standard NER classes	AIDA
Our EL baseline	88.8
w/ NER-enhanced Representation (NER-R)	89.0
w/ NER-enhanced Candidate Gen. (NER-CG)	89.1
w/ NER-based Negative Sampling (NER-NS)	89.1
w/ NER-Constrained Decoding (NER-CD)	90.7

Table 5: Results of our NER contributions on the AIDA-YAGO-CoNLL test set when using the four standard NER classes, i.e., PERSON, LOCATION, ORGANIZATION and MISCELLANEOUS.

more NER-based approaches brings further improvements, totaling a net gain of 3.9% of absolute improvement in average accuracy (or a 14% reduction in error rate) with respect to our already competitive EL baseline. While our main objective is not to propose a state-of-the-art model for EL, we observe that the application of NER is particularly beneficial on ACE2004, where our NER-enhanced EL system attains state-of-the-art results – 91.3% in accuracy compared to 91.2% of Fang et al. (2019) – trained only on the 18K sentences of AIDA-YAGO-CoNLL.

4.4 Analysis

One could argue that our NER-based contributions could still be effective with the four standard NER classes, i.e., Person, Organization, Location and Miscellaneous. However, as we can see from Table 5, using the standard NER classes greatly reduces the benefits of our NER-based contributions, especially due to the fact that the Miscellaneous class conflates several heterogeneous entity types into a single cluster.

In Table 6, instead, we report the per-class accuracy of our best system compared to our baseline, showing where our NER-based contributions bring more improvements. In general, our contributions positively affect each class, in particular ANIM (+33.3% in accuracy), FOOD (+66.7%) and TIME (+20.0%), i.e., our contributions help to correctly classify instances that are more difficult or rare.

NER class	Baseline	+NER	Δ
PER	95.8	96.5	+0.7
ORG	81.7	89.3	+7.6
LOC	93.4	94.3	+0.9
ANIM	66.7	100.0	+33.3
EVE	42.4	51.8	+9.4
FOOD	0.0	66.7	+66.7
INST	100.0	100.0	+0.0
MEDIA	90.0	95.0	+5.0
MON	100.0	100.0	+0.0
NUM	100.0	100.0	+0.0
PLANT	80.0	80.0	+0.0
SUPER	64.7	70.6	+5.9
TIME	60.0	80.0	+20.0
VEHI	86.7	100.0	+13.3

Table 6: Per-class accuracy of the entities in the AIDA-YAGO-CoNLL test set. Our NER-based contributions increase the performance of the baseline system on each entity class, especially the most difficult ones.

Finally, in Table 7 we provide a qualitative look at a few examples in which our NER-based contributions aid the EL system in choosing the correct named entity.

5 Conclusion and Future Work

In recent years, Entity Linking systems based on contextualized embeddings from pretrained language models have achieved unprecedented results. However, such systems require training on millions of labeled samples, making them practically inaccessible to broad audiences and users and severely hampering the development of a high-performance

Sentence	NER class	Prediction
Japan then laid siege to the Syrian penalty area for most of the game but rarely breached the **Syrian** defence.	ORG	**Baseline:** Syria **+NER**: Syria national football team
"You will not win the war of the Polish beer market with imported international brands", van Boxmeer said, adding that **Heineken** would remain an up-market import in Poland.	FOOD	**Baseline:** Heineken N.V. **+NER**: Heineken
Zieleniec led calls for the party and its leadership to listen to more diverse opinions, a thinly-veiled criticism of Klaus who has spearheaded the country's **post-Communist** economic reforms.	TIME	**Baseline:**: Democratic Left Alliance **+NER**: Post-communism
If successful the changes could get incorporated into future **Mars** missions Spirit and Opportunity were also fitted with a new navigation system that allows them to think several steps ahead.	CEL	**Baseline:**: Mars Pathfinder **+NER**: Mars
The five breeds credited with the most incidents were chow chows, Rottweilers, German shepherds, cocker spaniels and **Dalmatians**.	ANIM	**Baseline:**: Dalmatian Action **+NER**: Dalmatian (dog)

Table 7: Examples of sentences where the NER contributions help avoid errors. "Baseline" and "+NER" stand for the baseline system and the NER-enhanced best performing system, respectively.

EL system. In this paper, instead, we presented various NER-based strategies which allow systems trained on limited amounts of data to narrow the performance gap with those systems trained on massive training corpora. To this end, we first introduced a new fine-grained set of NER classes to better cluster entities and then used these classes to enhance a strong EL baseline with i) NER-enriched entity representations, ii) NER-enhanced candidate selection, iii) NER-based negative sampling, and iv) NER-constrained decoding. Our experiments show that the integration of NER information can aid an EL system trained on less than 20K instances in narrowing the gap with EL systems trained on millions of samples.

Over the past few years, the field of NER has witnessed continuous growth, with many researchers studying more complex forms of NER, including *nested* and *structured* NER (Finkel and Manning, 2009; Ju et al., 2018; Straková et al., 2019; Qian et al., 2020). Although we focused on the benefits of traditional NER in EL, we trust that more complex forms of NER can lead to even greater improvements in EL.

In conclusion, we believe that our work can encourage further developments on Entity Linking systems that require fewer and fewer training instances and still achieve strong results across in-domain and out-of-domain evaluations.

Acknowledgments

The authors gratefully acknowledge the support of the ERC Consolidator Grant MOUSSE No. 726487 under the European Union's Horizon 2020 research and innovation programme.

References

Jan A. Botha, Zifei Shan, and Daniel Gillick. 2020. Entity Linking in 100 Languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7833–7845, Online. Association for Computational Linguistics.

Samuel Broscheit. 2019. Investigating entity knowledge in BERT with simple neural end-to-end entity linking. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 677–685, Hong Kong, China. Association for Computational Linguistics.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval.

Marco De Gemmis, Pasquale Lops, Cataldo Musto, Fedelucio Narducci, and Giovanni Semeraro. 2015. Semantics-aware content-based recommender systems. In *Recommender systems handbook*, pages 119–159. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tommaso Di Noia and Vito Claudio Ostuni. 2015. Recommender systems and linked open data. In *Reasoning Web International Summer School*, pages 88–113. Springer.

Mohnish Dubey, Debayan Banerjee, Debanjan Chaudhuri, and Jens Lehmann. 2018. Earl: Joint entity and relation linking for question answering over knowledge graphs. In *The Semantic Web – ISWC 2018*, pages 108–126, Cham. Springer International Publishing.

Zheng Fang, Yanan Cao, Dongjie Zhang, Qian Li, Zhenyu Zhang, and Yanbing Liu. 2019. Joint entity linking with deep reinforcement learning.

Jenny Rose Finkel and Christopher D. Manning. 2009. Nested named entity recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 141–150, Singapore. Association for Computational Linguistics.

Evgeniy Gabrilovich, Michael Ringgaard, and Amarnag Subramanya. 2013. Facc1: Freebase annotation of clueweb corpora, version 1 (release date 2013-06-26, format version 1, correction level 0).

Octavian-Eugen Ganea and Thomas Hofmann. 2017. Deep joint entity disambiguation with local neural attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2619–2629, Copenhagen, Denmark. Association for Computational Linguistics.

Stephen Guo, Ming-Wei Chang, and Emre Kiciman. 2013. To link or not to link? a study on end-to-end tweet entity linking. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1020–1030, Atlanta, Georgia. Association for Computational Linguistics.

Zhaochen Guo and Denilson Barbosa. 2017. Robust named entity disambiguation with random walks. *Semantic Web*, 9:1–21.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 782–792, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Heng Ji and Ralph Grishman. 2011. Knowledge base population: Successful approaches and challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1148–1158, Portland, Oregon, USA. Association for Computational Linguistics.

Meizhi Ju, Makoto Miwa, and Sophia Ananiadou. 2018. A neural layered model for nested named entity recognition. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1446–1459, New Orleans, Louisiana. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. 2018. End-to-end neural entity linking. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 519–529, Brussels, Belgium. Association for Computational Linguistics.

Phong Le and Ivan Titov. 2018. Improving entity linking by modeling latent relations between mentions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1595–1604, Melbourne, Australia. Association for Computational Linguistics.

Phong Le and Ivan Titov. 2019. Boosting entity linking performance by leveraging unlabeled documents. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1935–1945, Florence, Italy. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chen Liang, Yue Yu, Haoming Jiang, Siawpeng Er, Ruijia Wang, Tuo Zhao, and Chao Zhang. 2020. Bond: Bert-assisted open-domain named entity recognition with distant supervision. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '20, page 1054–1064, New York, NY, USA. Association for Computing Machinery.

Thomas Lin, Mausam, and Oren Etzioni. 2012. Entity linking at web scale. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*, pages 84–88, Montréal, Canada. Association for Computational Linguistics.

Gang Luo, Xiaojiang Huang, Chin-Yew Lin, and Zaiqing Nie. 2015. Joint entity recognition and disambiguation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 879–888, Lisbon, Portugal. Association for Computational Linguistics.

Pedro Henrique Martins, Zita Marinho, and André F. T. Martins. 2019. Joint learning of named entity recognition and entity linking. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 190–196, Florence, Italy. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity linking meets word sense disambiguation: a unified approach. *Transactions of the Association for Computational Linguistics*, 2:231–244.

David Mueller, Nicholas Andrews, and Mark Dredze. 2020. Sources of transfer in multilingual named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8093–8104, Online. Association for Computational Linguistics.

Cataldo Musto, Giovanni Semeraro, Pasquale Lops, and Marco de Gemmis. 2014. Combining distributional semantics and entity linking for context-aware content-based recommendation. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 381–392. Springer.

Roberto Navigli, Michele Bevilacqua, Simone Conia, Dario Montagnini, and Francesco Cecconi. 2021. Ten years of BabelNet: A survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4559–4567. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217 – 250.

Dat Ba Nguyen, Martin Theobald, and Gerhard Weikum. 2016. J-NERD: Joint named entity recognition and disambiguation with rich linguistic features. *Transactions of the Association for Computational Linguistics*, 4:215–229.

Laurel Orr, Megan Leszczynski, Simran Arora, Sen Wu, Neel Guha, Xiao Ling, and Christopher Re. 2020. Bootleg: Chasing the tail with self-supervised named entity disambiguation.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2021. Kilt: a benchmark for knowledge intensive language tasks.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea. Association for Computational Linguistics.

Kun Qian, Poornima Chozhiyath Raman, Yunyao Li, and Lucian Popa. 2020. Learning structured representations of entity names using ActiveLearning and weak supervision. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6376–6383, Online. Association for Computational Linguistics.

Jonathan Raiman and Olivier Raiman. 2018. Deeptype: Multilingual entity linking by neural type system evolution.

Delip Rao, Paul McNamee, and Mark Dredze. 2013. *Entity Linking: Finding Extracted Entities in a Knowledge Base*, pages 93–115. Springer Berlin Heidelberg, Berlin, Heidelberg.

Hamed Shahbazi, Xiaoli Z. Fern, Reza Ghaeini, Rasha Obeidat, and Prasad Tadepalli. 2019. Entity-aware elmo: Learning contextual entity representation for entity disambiguation.

Fábio Souza, Rodrigo Nogueira, and Roberto Lotufo. 2020. Portuguese named entity recognition using bert-crf.

Jana Straková, Milan Straka, and Jan Hajic. 2019. Neural architectures for nested NER through linearization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5326–5331, Florence, Italy. Association for Computational Linguistics.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, page 697–706, New York, NY, USA. Association for Computing Machinery.

Simone Tedeschi, Valentino Maiorca, Niccolò Campolungo, Francesco Cecconi, and Roberto Navigli. 2021. WikiNEuRal: Combined neural and knowledge-based silver data creation for multilingual NER. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP 2021)*, Punta Cana, Dominican Republic.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Antti Virtanen, Jenna Kanerva, Rami Ilo, Jouni Luoma, Juhani Luotolahti, Tapio Salakoski, Filip Ginter, and Sampo Pyysalo. 2019. Multilingual is not enough: Bert for finnish.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Xiyuan Yang, Xiaotao Gu, Sheng Lin, Siliang Tang, Yueting Zhuang, Fei Wu, Zhigang Chen, Guoping Hu, and Xiang Ren. 2019. Learning dynamic context augmentation for global entity linking. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 271–281, Hong Kong, China. Association for Computational Linguistics.

Wenpeng Yin, Mo Yu, Bing Xiang, Bowen Zhou, and Hinrich Schütze. 2016. Simple question answering by attentive convolutional neural network. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1746–1756, Osaka, Japan. The COLING 2016 Organizing Committee.

A Hardware and Training Details

All model training was carried out on a NVIDIA GeForce RTX 3090. It required $\sim$2h/epoch on the AIDA-YAGO-CoNLL training set, for an average number of $\sim$20 epochs.

B NER Classes

In this Section we provide further details about the set of NER classes we used. We designed our set of classes starting from the 18 fine-grained classes used for OntoNotes 5.0, splitting and merging them to better fit the EL task.

In Table 9 we show the mapping between our classes and the OntoNotes ones. For example, we split the PRODUCT class of OntoNotes into three separate classes, namely, FOOD, INST and VEHI. These three classes are very different from each other, and a NER classifier will easily predict the correct one, so keeping them separated helps in better clustering entities. On the other hand, they use 3 different tags (LOC, FAC and GPE) to represent our LOC class, but in this case their 3 classes are very similar, and a NER classifier could easily get confused. Similarly, they use 4 classes QUANTITY, ORDINAL, CARDINAL and PERCENT to express our NUM class. Again, distinguishing between these classes is hard and, in this case, even useless for our task. Finally, 6 out of our 18 classes, which are useful for better distinguishing entities, do not have a corresponding class in the OntoNotes categorization. Then, in Table 10 we report a textual description for each of the considered classes, both the OntoNotes ones and our classes. Specifically, in the top part of the table we show the 18 classes of OntoNotes, whereas in the bottom part we show those of the subset of our classes which need a separate description. For instance, we describe our ANIM class because it does not have a corresponding class in OntoNotes, but we do not describe our LOC class because we know from Table 10 that it corresponds to the three classes LOC, FAC and GPE of OntoNotes.

C NER-constrained Decoding

Soft and Hard Constraints. In the Section about the contribution of NER-constrained decoding, we introduced soft and hard constraints. In this Section instead, we show how these constraints affect the final performance of the complete EL + NER system. In Table 11 we report the results obtained. In the second row of the table we have the result with the hard constraint (i.e., 91.7), namely, the class of a given candidate must exactly match the predicted one in order to be considered. In the following four rows we relax this constraint, and we impose the constraint that the predicted class, to be considered reliable (i.e., to actually filter candidates with a different type), must be above a certain threshold. The higher the threshold the more accurate the prediction is, but the lower the number of mentions considered is. The best results are achieved using a confidence threshold of 0.5. This means that considering also classes predicted with low confidence (< 0.5) introduces errors, while using a higher threshold decreases the number of applications of our technique. In the second block of the table instead, we keep only candidates whose type is within the top-k types predicted by the NER classifier. The higher the value of k is, the lower the probability of discarding the target entity is, but less the size of the candidate set is reduced. We observe that considering the top-k

Model	AIDA	MSNBC	AQUAINT	ACE2004	CWEB	WIKI	Avg.
Our EL baseline	88.8	86.9	61.8	89.9	65.7	58.5	75.3
w/ NER-Constrained Decoding Jointly-Learnt (NER-CD-JL)	90.4	87.9	66.9	90.5	67.1	62.0	77.5
w/ NER-Constrained Decoding (NER-CD)	92.2	89.2	68.2	91.2	68.1	63.8	78.8

Table 8: InKB accuracy of the baseline system and of the two variants of the NER-Constrained Decoding contribution on the in-domain and out-of-domain test sets.

Our Class	OntoNotes Class
PER	PERSON
ORG	ORG, NORP
LOC	LOC, FAC, GPE
ANIM	-
BIO	-
CEL	-
DIS	-
EVE	EVENT
FOOD	PRODUCT
INST	PRODUCT
MEDIA	WORK_OF_ART, LANGUAGE
MON	MONEY
NUM	QUANTITY, ORDINAL, CARDINAL, PERCENT
PHY	EVE
PLANT	-
SUPER	-
TIME	DATE, TIME
VEHI	PRODUCT
-	LAW

Table 9: Comparison between our new set of fine-grained NER classes and the OntoNotes ones.

Our Class	Description
PERSON	People, including fictional characters
NORP	Nationalities or religious or political groups
ORG	Companies, agencies, institutions, etc.
FAC	Buildings, airports, highways, bridges, etc.
GPE	Countries, cities, states
LOC	Non-GPE locations, mountain ranges, bodies of water
PRODUCT	Objects, vehicles, foods, etc. (not services)
EVENT	Named hurricanes, battles, wars, sport events, etc.
WORK_OF_ART	Titles of books, songs, etc.
LAW	Named documents made into laws
LANGUAGE	Any named language
DATE	Absolute or relative dates or periods
TIME	Times smaller than a day
PERCENT	Percentages, including "%"
MONEY	Monetary values, including unit
QUANTITY	Measurements, as of weight or distance
ORDINAL	"first", "second", etc.
CARDINAL	Numerals that do not fall under another type
ANIM	Breeds of dogs, cats and other animals
BIO	Genes, proteins and other biological entities
CEL	Planets, stars, asteroids and other celestial bodies
DIS	Named diseases
FOOD	Foods, drinks, etc.
INST	Technical instruments, musical instruments, etc.
PHY	Named hurricanes and other physical phenomena
PLANT	Types of trees, flowers, etc.
SUPER	Supernatural entities
VEHI	Car models, motorcycle models, etc.

Table 10: Textual description for each NER class.

classes is not as good considering only the most probable one. From the third block of the table onwards, we combine confidence thresholds and top-k classes. This strategy allows us to further improve performances. In particular, we obtain the best results using a threshold of 0.5 and k=3, so we first check if the classifier predicted the class with a confidence > 0.5, and: i) if this is the case, we consider only the most probable class, otherwise, ii) we switch to the top-3 predicted classes.

Joint-learnt NER classifier. In Section 3.6, we stated that for the NER classifier it is possible to train a separate model, or train the EL system not only to assign the most appropriate entity to a mention in context, but also to provide its NER class. The advantage of the second approach is that it requires a single model and, therefore, fewer computational resources. However, performing both tasks jointly leads to worse results in NER labeling, which, in turn, diminishes the benefits of our NER-constrained decoding strategy for the overall EL system, as shown in Table 8.

Model	Confidence	Top-k	Accuracy AIDA
Our Baseline	-	-	88.8
Our Approach + NER filter	0.00	-	91.7
	0.50	-	91.9
	0.80	-	91.7
	0.90	-	91.3
	0.99	-	90.0
	-	k=2	90.6
	-	k=3	90.0
	-	k=4	89.5
	-	k=5	89.1
	0.50	k=2	92.1
	0.80	k=2	91.9
	0.90	k=2	91.7
	0.99	k=2	91.2
	0.50	k=3	**92.2**
	0.80	k=3	92.0
	0.90	k=3	91.7
	0.99	k=3	91.4
	0.50	k=4	91.9
	0.80	k=4	91.6
	0.90	k=4	91.2
	0.99	k=4	90.9
	0.50	k=5	91.8
	0.80	k=5	91.6
	0.90	k=5	91.2
	0.99	k=5	90.8

Table 11: Performance of the NER-constrained decoding contribution with soft and hard constraints.